REPORT NO: NAWCADWAR--96-21-TR

DETERMINING THE RANK OF A NOISY SQUARE MATRIX USING THE CHARACTERISTIC COEFFICIENTS

Ronald F. Gleeson, Ph.D.
Department of Physics
Trenton State College
Trenton, NJ 08650

Peter F. Stiller, Ph.D.
Department of Mathematics
Texas A & M University
College Station, TX 77843-3368

Robert M. Williams, Ph.D.
Mission Avionics Technology Dept. (Code 455100R07)
Naval Air Systems Team
Warminster, PA 18974-5000

15 JULY 1996

FINAL REPORT
Period Covering June 1995 to July 1996

Prepared for
OFFICE OF NAVAL RESEARCH
800 N. Quincy Street
Arlington, VA 22217

PRODUCT ENDORSEMENT - The discussion or instructions concerning commercial products herein do not constitute an endorsement by the Government nor do they convey or imply the license or right to use such products.

Reviewed By: _____ **Date:** 7/18/96
Author/COTR

Reviewed By: _____ **Date:** 7/25/96
Author/COTR

Reviewed By: _____ **Date:** 7/26/96
Author/COTR

Released By: _____ **Date:** 7/29/96
Level III Manager

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	15 July 1996	June 1995 to July 1996

4. TITLE AND SUBTITLE
Determining the Rank of a Noisy Square Matrix Using the Characteristic Coefficients

5. FUNDING NUMBERS

6. AUTHOR(S)
*Ronald F. Gleeson, Ph.D. **Peter F. Stiller, Ph.D.
Robert M. Williams, Ph.D.

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Naval Air Warfare Center
Aircraft Division Warminster
Code 455100R07
Warminster, PA 18974-0591

8. PERFORMING ORGANIZATION REPORT NUMBER
NAWCADWAR--96-21-TR

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Office of Naval Research
800 N. Quincy Street
Arlington, VA 22217

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES
*Department of Physics
Trenton State College
Trenton, NJ 08650

**Department of Mathematics
Texas A & M University
College Station, TX 77843-3368

12a. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for Public Release; Distribution is Unlimited

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

Determining the rank of a matrix has several important applications. In modern array processing, the rank can be used to determine the number of targets detected. Radar normally performs this operation prior to determining the direction and velocity of each air platform. Also, sonar performs this operation before it attempts to classify submarines. It is therefore essential that the rank of the signal matrix be determined efficiently and accurately. It is assumed that the signal matrix is square and free of a nilpotent part. Unfortunately, there is usually noise added to the elements of the signal matrix due to such factors as the background in which the signal is embedded or instrument uncertainty.

This paper describes a method for predicting the rank of a signal matrix by analyzing the coefficients of the characteristic polynomial of the noisy version of this matrix. These coefficients can be computed from algebraic sums of products of the elements. Time consuming iterations, which take place in methods involving singular values, are avoided. Furthermore, the results are shown to compare favorably to those produced by a singular value approach.

14. SUBJECT TERMS
Matrix rank, Noisy data, Array processing.

15. NUMBER OF PAGES
37

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	UL

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

Abstract

Determining the rank of a matrix has several important applications. In modern array processing, the rank can be used to determine the number of targets detected. Radar normally performs this operation prior to determining the direction and velocity of each air platform. Also, sonar performs this operation before it attempts to classify submarines. It is therefore essential that the rank of the signal matrix be determined efficiently and accurately. It is assumed that the signal matrix is square and free of a nilpotent part. Unfortunately, there is usually noise added to the elements of the signal matrix due to such factors as the background in which the signal is embedded or instrument uncertainty.

This paper describes a method for predicting the rank of a signal matrix by analyzing the coefficients of the characteristic polynomial of the noisy version of this matrix. These coefficients can be computed from algebraic sums of products of the elements. Time consuming iterations, which take place in methods involving singular values, are avoided. Furthermore, the results are shown to compare favorably to those produced by a singular value approach.

Determining the Rank of a Noisy Square Matrix Using the Characteristic Coefficients

I. Introduction

An NxN real matrix M is a representation of a linear operator from R^N onto a certain subspace called the column space of M. The number of linearly independent vectors needed to span the column space is N or less and is called the rank of the matrix. For example, if a 7x7 matrix only needed four vectors to span its column space, then the rank would be four. This rank of four would manifest itself by M having only four nonzero eigenvalues or four nonzero singular values.

Determining the rank of a matrix has several important applications. In modern array processing, the rank can be used to determine the number of targets detected. Radar normally performs this operation prior to determining the direction and velocity (using the doppler shift) of each air platform. Also, sonar performs this operation before it attempts to classify submarines. It is therefore essential that the rank of the signal matrix be determined efficiently and accurately. It will be assumed throughout this paper that the signal matrices we are discussing are square and do not have a nilpotent part. Unfortunately, there is usually noise added to the elements of the signal matrix due to such factors as the background in which the signal is embedded or instrument uncertainty.

This report describes a Monte Carlo method for determining the rank of a signal matrix using the coefficients of the characteristic polynomial of the noisy version of this matrix. These coefficients will be referred to as the C_k's. The Monte Carlo calculations are done in the lab. The matrix determined in the field is then compared with these Monte Carlo results. Finally, the success rate of the model is discussed for different levels of noise and compared with a singular value approach.

II. The Characteristic Coefficients

A. Computing the Characteristic Coefficients

Recall that every matrix has a characteristic equation. This equation is formed by: (1) subtracting x from each element on the diagonal of the matrix, (2) taking the determinant of the new matrix, and (3) setting this determinant equal to zero. For example, consider the 2x2 matrix below:

$$M = \begin{pmatrix} A1 & A2 \\ B1 & B2 \end{pmatrix} \tag{2.1}$$

Subtracting x from each of the diagonal elements and setting the determinant equal to zero, we get:

$$\begin{vmatrix} A1 - x & A2 \\ B1 & B2 - x \end{vmatrix} = 0 \tag{2.2}$$

This yields the equation:

$$(A1 - x)(B2 - x) - (A2 \times B1) = 0 \tag{2.3}$$

Expanding equation (2.3) produces:

$$x^2 - (A1 + B2)x + (A1 \times B2 - A2 \times B1) = 0 \tag{2.4}$$

Equation (2.4) is the characteristic equation, and it can be put into the form:

$$x^2 + C_1 x + C_2 = 0 \tag{2.5}$$

In equation (2.5), C_1 and C_2 are the characteristic coefficients (the C_k's for k = 1 and 2) and are equal to:

$$C_1 = -(A1 + B2) \tag{2.6}$$

$$C_2 = (A1 \times B2 - A2 \times B1) \tag{2.7}$$

C_1 is the negative of the trace of matrix M, and C_2 is the determinant of M.

For the 3x3 case, the characteristic equation comes from the following determinant set equal to zero:

$$(-1)^3 \begin{vmatrix} (A1-x) & A2 & A3 \\ B1 & (B2-x) & B3 \\ C1 & C2 & (C3-x) \end{vmatrix} = 0 \tag{2.8}$$

The characteristic equation is:

$$x^3 + C_1 x^2 + C_2 x + C_3 = 0 \tag{2.9}$$

where

$$C_1 = -(A1 + B2 + C3) \tag{2.10}$$

$$C_2 = +(A1B2 - A2B1 + A1C3 - A3C1 + B2C3 - B3C2) \tag{2.11}$$

$$C_3 = -(A1B2C3 + A2B3C1 + A3B1C2 - A3B2C1 - A1B3C2 - A2B1C3) \tag{2.12}$$

C_1 again is the negative of the trace of the matrix, and C_3 is now the negative of the determinant. C_2 is the sum of three 2x2 determinants. The first of the 2x2 determinants is formed using the four elements defined by the intersection of the first two rows with the first two columns. The second determinant is the intersection of the first and third rows and columns, and the last determinant is the intersection of the second and third rows and columns. In short, C_2 is the sum of all determinants that can be formed by taking the intersection of two rows with the same two columns. Up to a sign this is what happened with C_1 and C_3 also. C_3 is the negative of the sum of all determinants involving three rows with the same three columns. In the case of a 3x3 matrix there is only one such combination; however, for a larger matrix there will be many such

combinations. For C_1 each of the determinants clearly involves only one element.

In general, for an NxN matrix, C_k is formed by multiplying minus one to the kth power times the sum of all the determinants that can be formed from the N rows and columns taken k at a time. The k rows must correspond to the same k columns. For example, suppose we had a 7x7 matrix. One of the determinants for C_3 would involve rows 1, 3 and 6. That determinant must also involve columns 1, 3, and 6. There would be 7 choose 3 (which calculates to 35) such determinants in the sum for C_3.

In short, each of the C_k's can be written down as of a formula involving an algebraic sum of products of the elements of the original matrix. This fact allows for parallel processing in the determination of the C_k's. In particular, it is important to note that their calculation does not involve root finding or the sort of iterative computation which usually characterizes the computation of eigenvalues and singular values.

B. Useful Properties of the Characteristic Coefficients (the C_k's)

While we are recommending that the C_k's be actually computed in terms of the elements of the original matrix as described above, there is another way to compute them that demonstrates a useful property of the C_k's. The useful property is that if we have, for example, a 7x7 matrix with a rank of 4 (N.B. we are assuming no nilpotent part throughout this report), then C_5, C_6 and C_7 are all zero. Furthermore, C_4 is nonzero. In general, for a NxN matrix of rank M, the last N minus M C_k's are zero. This statement is most easily shown by considering the calculation of the C_k's in terms of the eigenvalues (λ's). The C_k's are invariant under a similarity transformation. Assuming (as we have) that the matrix can be put into diagonal form, the diagonal elements are the λ's. The C_k's can be calculated for the diagonal matrix using the same rules which were discussed above. In particular, C_2 would still be the sum of the N choose 2 determinants that can be formed. The determinants

each only involve one term since the matrix is diagonal. For example, for a 3x3 matrix,

$$C_2 = (\lambda_1 \lambda_2 + \lambda_1 \lambda_3 + \lambda_2 \lambda_3) \qquad (2.13)$$

where $\lambda_1 \lambda_2$ is the determinant for rows and columns 1 and 2. Looking at the set of all three C_k's for the 3x3 matrix in diagonal form:

$$C_1 = -(\lambda_1 + \lambda_2 + \lambda_3) \qquad (2.14)$$
$$C_2 = (\lambda_1 \lambda_2 + \lambda_1 \lambda_3 + \lambda_2 \lambda_3) \qquad (2.15)$$
$$C_3 = -\lambda_1 \lambda_2 \lambda_3 \qquad (2.16)$$

We see easily that the number of nonzero eigenvalues equals the rank of the matrix. For the 3x3 matrix if only one of the eigenvalues is zero (and therefore two are nonzero), then the rank is two; however, expressing the C_k's in terms of the λ's one can see that if only one of the eigenvalues is zero, then C_3 is zero, while C_2 is nonzero. If the rank is one, and two of the eigenvalues are zero, then C_2 and C_3 are both zero, while C_1 is nonzero . Finally, if the rank is zero because all three eigenvalues are zero, then all three of the C_k's are zero.

If we were working with a 7x7 matrix and the rank were six, then one λ would be zero, and this would cause C_6 to be nonzero and C_7 to be zero. If the rank of the 7x7 matrix were 4, then C_4 would be nonzero and C_5, C_6 and C_7 would all be zero. Determining which C_k's are zero, in theory, tells us the rank of the diagonalizable matrix.

C. The Problem with the C_k's

Most data is contaminated by noise The problem then is to estimate the rank in the presence of noise. This noise also affects the C_k's. For example, a 7x7 matrix of rank 4 in the presence of noise will in general not have C_5, C_6 and C_7 equal to zero.

In order to appreciate the extent of this problem consider the following diagonalizable 7x7 matrix M with rank 4 (not obvious):

$$M = \begin{pmatrix} -11.803 & -9.679 & 9.459 & -4.238 & 6.708 & -10.425 & -7.105 \\ 49.566 & -13.561 & 82.518 & 3.085 & -24.717 & -51.103 & 9.505 \\ 33.564 & 14.296 & 3.511 & 8.375 & -18.367 & 5.907 & 15.057 \\ 45.290 & 2.359 & 25.720 & 7.981 & -21.422 & -5.088 & 18.353 \\ 30.278 & -19.164 & 57.783 & -1.179 & -13.972 & -34.612 & 4.884 \\ 5.826 & 33.312 & -67.536 & 8.888 & -6.442 & 53.129 & 13.097 \\ 5.591 & 29.720 & -36.245 & 12.981 & -0.088 & 28.777 & 11.715 \end{pmatrix} \quad (2.17)$$

The C_k's of M compute to be the set: $\{C_1,C_2,C_3,C_4,C_5,C_6,C_7\}$ = {-37, -109, 11317, -41412, 0, 0, 0}.

If noise at the 0.1 level is added, we get the matrix M' below. The details of how M was generated and how the noise was added to form M' are explained in the next chapter.

$$M' = \begin{pmatrix} -11.867 & -9.567 & 9.276 & -4.174 & 6.577 & -10.404 & -6.950 \\ 49.453 & -13.434 & 82.401 & 3.136 & -24.866 & -51.159 & 9.391 \\ 33.591 & 14.290 & 3.507 & 8.537 & -18.381 & 5.797 & 14.882 \\ 45.263 & 2.392 & 25.537 & 8.019 & -21.382 & -5.172 & 18.378 \\ 30.248 & -19.050 & 57.955 & -1.217 & -13.951 & -34.529 & 4.831 \\ 5.760 & 33.358 & -67.592 & 8.979 & -6.405 & 53.054 & 13.205 \\ 5.642 & 29.668 & -36.238 & 13.045 & -0.068 & 28.822 & 11.821 \end{pmatrix} \quad (2.18)$$

The C_k's for M' are now changed to the set: {-37.1, -106.0, 11407.0, -40571.9, -3015.2, -30003.3, -6525.5}. All the C_k's are changed, but most importantly, C_5, C_6 and C_7 are far from zero. Moreover, if M' were to be multiplied through by a factor of 10 things would get worse. C_1 would be 10 times larger; C_2 would be 100 times larger because it involves products of two elements at a time; C_3 would be 1000 times larger, etc. In this case C_5, C_6 and C_7 would be even more enormous.

D. The Solution to the C_k Problem

To solve this problem the matrix needs to be normalized by dividing through by a constant so that the normalized eigenvalues would tend to be less than one. The normalizing factor (N.F.) should be easy to compute, tend not to be zero, and reflect the size of the eigenvalues. We chose the following N.F.:

N.F. = square root of the the absolute value of the sum of the squares of the eigenvalues (2.19)

The sum of the squares of the eigenvalues happens to be equal to the trace of the matrix squared. To understand this last statement, express the matrix in diagonal form, multiply it by itself, and then take its trace. We called this trace t_2.

$$N.F. = \sqrt{|t_2|} \tag{2.20}$$

It turns out that t_2 can also be computed from the C_k's:

$$t_2 = C_1^2 - 2C_2 \tag{2.21}$$

Instead of dividing all the matrix elements by N.F., we normalized the C_k's themselves. This was done by creating a normalized set of coefficients which we called the P_k's, where:

$$P_k = \frac{\sqrt[k]{|C_k|}}{\sqrt{|t_2|}} = \frac{\sqrt[k]{|C_k|}}{N.F} \tag{2.22}$$

When two or more of the eigenvalues are complex, it is possible (albeit extremely rare) that t_2 is zero. Since the computation of the P_k's involves a division by t_2, a check to see if t_2 is zero should be done before this method is used.

III. Description of Our Monte Carlo Simulation

A. Generating Noisy Matrices

We developed and tested our approach by studying 7x7 matrices using Mathematica. For each rank from one to seven we generated 1000 matrices with noise added at a prescribed level. This procedure of generating the matrices and adding the noise will be described in detail in succeeding paragraphs. The set of seven P_k's for each matrix was computed. The effect of rank on the P_k distributions was studied.

To generate a matrix with a particular rank we first chose an appropriate set of eigenvalues. We then used these eigenvalues to create a diagonal matrix. After this, we performed a similarity transformation on the diagonal matrix to put it into a more general form. Finally, we added the noise.

Next we shall explain the above process in more detail using as an example the 7x7 matrix M discussed in the previous chapter:

$$M = \begin{pmatrix} -11.803 & -9.679 & 9.459 & -4.238 & 6.708 & -10.425 & -7.105 \\ 49.566 & -13.561 & 82.518 & 3.085 & -24.717 & -51.103 & 9.505 \\ 33.564 & 14.296 & 3.511 & 8.375 & -18.367 & 5.907 & 15.057 \\ 45.290 & 2.359 & 25.720 & 7.981 & -21.422 & -5.088 & 18.353 \\ 30.278 & -19.164 & 57.783 & -1.179 & -13.972 & -34.612 & 4.884 \\ 5.826 & 33.312 & -67.536 & 8.888 & -6.442 & 53.129 & 13.097 \\ 5.591 & 29.720 & -36.245 & 12.981 & -0.088 & 28.777 & 11.715 \end{pmatrix} \quad (3.1)$$

The rank of the matrix was determined by controlling the number of nonzero eigenvalues. If we wanted a rank 4 matrix, we would choose 4 nonzero eigenvalues. The eigenvalues were randomly chosen from the set of integers from 1 to 30. For each eigenvalue we, in effect, tossed a coin to decide whether it should be positive or negative. The coin tossing was carried out by choosing a floating point number randomly from 0 to 1. If the floating point

number was less than 0.5, a minus sign was associated with the eigenvalue; otherwise, it remained positive. For our example matrix M, this set turned out to be: {4, 21, 29, -17}

A diagonal matrix (D) was created with the chosen set of eigenvalues placed on the diagonal and zeros padding the remaining positions. In the above 7x7 matrix of rank 4, the set of 4 eigenvalues and 3 zeros were positioned on the diagonal. In our example,

$$D = \begin{pmatrix} 4 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 21 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 29 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -17 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \tag{3.2}$$

Next, a similarity transformation, $M = TDT^{-1}$, was performed on the diagonal matrix to put the matrix into a more general form. Each element of the transformation T was separately randomly chosen from the range of floating point numbers from -1.0 to +1.0. The particular T which produced M from D was:

$$T = \begin{pmatrix} -0.058 & -0.318 & 0.340 & -0.231 & -0.718 & 0.561 & -0.364 \\ 0.669 & -0.275 & -0.898 & 0.656 & 0.553 & 0.391 & -0.222 \\ 0.333 & 0.493 & -0.836 & 0.576 & 0.974 & 0.276 & 0.257 \\ -0.833 & 0.635 & -0.896 & 0.940 & -0.940 & -0.724 & 0.883 \\ -0.354 & -0.127 & -0.263 & 0.723 & -0.224 & 0.721 & -0.587 \\ 0.267 & 0.899 & -0.383 & 0.334 & 0.919 & 0.390 & 0.384 \\ -0.569 & 0.681 & -0.991 & -0.664 & 0.737 & -0.557 & -0.396 \end{pmatrix} \tag{3.3}$$

Here the elements of T have been rounded to three decimal places.

Noise was handled in the following manner: Suppose noise at the level of 0.1 was to be added to the matrix M. This value (0.1) was then used as the standard deviation for the normal distribution with a mean of zero. Using Mathematica's intrinsic NormalDistribution function, a value of "x" was randomly chosen from this distribution and added to the first element value. In the actual calculation of our matrix M', the noisy version of M, the value -0.064 was chosen for "x," and the first element of M' became -11.803 + (-0.064) = -11.867 . The process was then repeated independently for each of the other elements to produce:

$$M' = \begin{pmatrix} -11.867 & -9.567 & 9.276 & -4.174 & 6.577 & -10.404 & -6.950 \\ 49.453 & -13.434 & 82.401 & 3.136 & -24.866 & -51.159 & 9.391 \\ 33.591 & 14.290 & 3.507 & 8.537 & -18.381 & 5.797 & 14.882 \\ 45.263 & 2.392 & 25.537 & 8.019 & -21.382 & -5.172 & 18.378 \\ 30.248 & -19.050 & 57.955 & -1.217 & -13.951 & -34.529 & 4.831 \\ 5.760 & 33.358 & -67.592 & 8.979 & -6.405 & 53.054 & 13.205 \\ 5.642 & 29.668 & -36.238 & 13.045 & -0.068 & 28.822 & 11.821 \end{pmatrix} \quad (3.4)$$

B. Using the P_k Profile to Determine the Rank

The 7 C_k's were then computed. For the above matrix M' this set turns out to be those listed in chapter II, namely, {-37.1, -106.0, 11407.0, -40571.9, -3015.2, -30003.3, -6525.5}. The normalizing factor (N.F.) is computed as:

$$t_2 = C_1^2 - 2C_2 \quad (3.5)$$

$$t_2 = (-37.1)^2 - 2(-106.0) \quad (3.6)$$

$$t_2 = 1588.41 \quad (3.7)$$

$$N.F. = \sqrt{|t_2|} \quad (3.9)$$

$$N.F. = \sqrt{|1588.41|} \quad (3.10)$$

$$N.F. = 39.85 \quad (3.11)$$

The set of 7 P_k's were then calculated from:

$$P_k = \frac{\sqrt[k]{|C_k|}}{\sqrt{|t_2|}} \tag{3.12}$$

For example P_3 would be:

$$P_3 = \frac{\sqrt[3]{|C_3|}}{N.F} \tag{3.13}$$

$$P_3 = \frac{\sqrt[3]{|11407.0|}}{39.85} \tag{3.14}$$

$$P_3 = 0.56 \tag{3.15}$$

The set of P_k's for the matrix M' above are:
{0.93, 0.26, 0.56, 0.36, 0.12, 0.14, 0.09}

To determine the rank, these values of the P_k's have to be compared against certain threshold values. Recall, if there were no noise, then for the example M, $C_5 = C_6 = C_7 = 0$ and C_4 is nonzero. With the noise level set to 0.1, these C_k's got rather large. The P_k's are much more robust. We expect P_5, P_6 and P_7 to be relatively low, and P_4 to be relatively high. How we determine the thresholds for sorting out low from high will be discussed in the next chapter. The thresholds when the noise level is 0.1 will be shown to range between 0.2 and 0.3. Values like those of P_5, P_6 and P_7, namely, {0.12, 0.14, 0.09} respectively, are low relative to thresholds in this range; whereas a P_4 value of 0.36 is relatively high.

This pattern of P_k's, a high P4 and low values of P_5, P_6 and P_7 we will refer to as the profile for rank 4. A matrix such as M' having this profile would be predicted to be rank 4. Keep in mind C_1, C_2 and C_3 and therefore P_1, P_2 and P_3 can be low or high for rank 4. Therefore, they cannot be used as part of the rank 4 profile.

The entire P_k profile set for a 7x7 matrix is the following:
rank 7 = high P_7;
rank 6 = high P_6, and low P_7;

rank 5 = high P_5, and low P_6 and P_7;
rank 4 = high P_4, and low P_5, P_6 and P_7;
rank 3 = high P_3, and low P_4, P_5, P_6 and P_7;
rank 2 = high P_2, and low P_3 P_4 P_5, P_6 and P_7;
rank 1 = high P1 and low P_2, P_3, P_4, P_5, P_6 and P_7

IV. Results of the Monte Carlo Simulation

A. Testing the Ability to Distinguish Ranks

The first thing we checked was the effect of rank on the critical P_k's. For example, since we were generating 7 X 7 matrices, P_7 was the critical P_k for distinguishing matrices with rank 7 from matrices of lower rank. Matrices with a rank less than 7 should have a somewhat lower value of P_7 than matrices with rank 7. Recall, in the noise free environment P_7 for matrices with a rank less than 7 should be zero. What is compared below are two histograms of P_7 values where the noise level was set at 0.1. The histogram with the large dots is for rank 7 matrices; the histogram with the small squares is for rank 6.

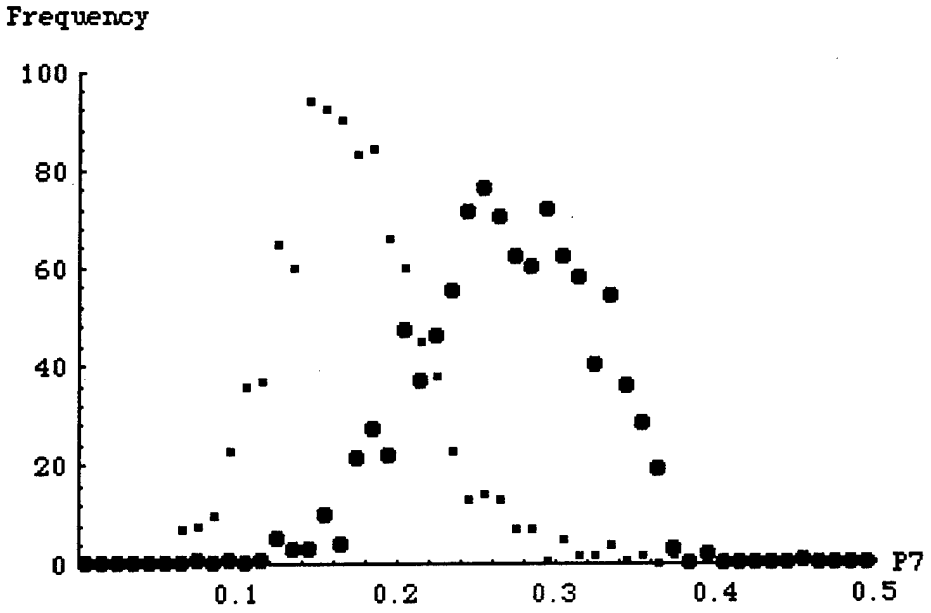

Figure 1. Frequency versus P_7 values for rank 6 (small squares) and rank 7 (large dots).

If a P_7 value of 0.21 is used as a threshold value, we find that 81% of the rank 6 matrices have their P_7 value lower than this value; whereas, 86% of the rank 7 matrices have their P_7 value higher than this value.

A prior study had been performed using percent noise. The standard deviation for the noise to be added to each term was taken to be a certain percentage of each element rather than an absolute value. When the noise was set at the one percent level, the same sort of separation between rank 6 and rank 7 occurred although the percentages were somewhat different. In this earlier study we also tested the other critical P_k's, namely P_6 to distinguish between rank 6 and rank 5, etc., and similar separations were found for the other critical P_k's.

B. Setting the Thresholds

In the latest study (using absolute values for the noise), 1000 matrices were generated for each rank from rank one to seven. The set of P_k's were calculated for each matrix. This process was performed with the absolute noise set at different values. Using this pool of data, we could examine the effect of setting the threshold levels to various values. As an example of what we did next, consider the effect of setting the noise level to 0.1 and the thresholds for the P_1 to P_7 to the values { 0.11, 0.29, 0.25, 0.25, 0.23, 0.23, 0.29 } respectively. How we arrived at these specific thresholds will be explained later. Recall, the profile for rank 1 was a P_1 value higher than its threshold (0.11), and P_2 through P_7 values lower than their thresholds of {0.29, 0.25, 0.25, 0.23, 0.23, 0.29}. This profile was satisfied by 777 (out of 1000) matrices whose actual rank was one. Below is a table of the number of matrices out of 1000 that had the rank 1 profile versus the actual rank.

Actual Rank	1	2	3	4	5	6	7	Total
Number of Matrices with Rank 1 profile	777	61	6	0	0	0	0	844
Percent of Total with Rank 1 profile	92.1	7.2	0.7	0	0	0	0	100

Table 1. Raw data with the noise level set at 0.1 and rank 1 profile

From this table we see that when the noise level is 0.1, the success rate for the profile for rank one is (777/844) x 100% = 92.1%. This kind of data was gathered for each of the seven profiles.

To improve our selection of the thresholds we first set all seven thresholds to 0.01 and computed the percent success rate for each profile. Then we incremented these synchronized thresholds to 0.02 and recomputed these success rates. We continued incrementing the thresholds by 0.01 over the range from 0.00 to 1.00, and then plotted the percent success rate for each profile versus the synchronized threshold value. A sample of four of these plots for profiles 1, 3, 5 and 7 appears below.

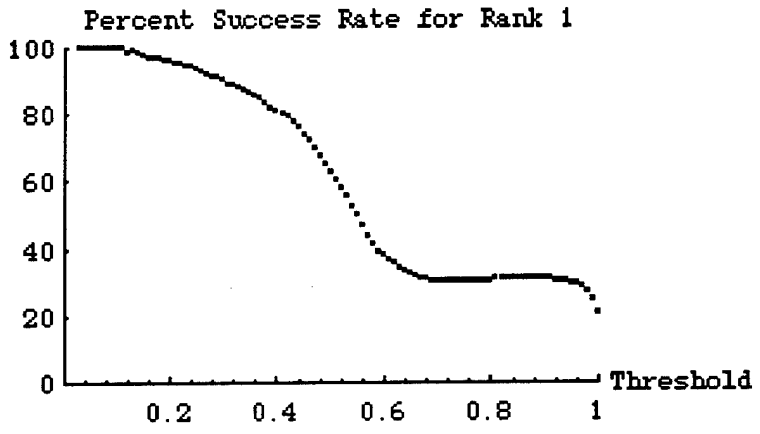

Figure 2. Percent success rates versus threshold for rank 1

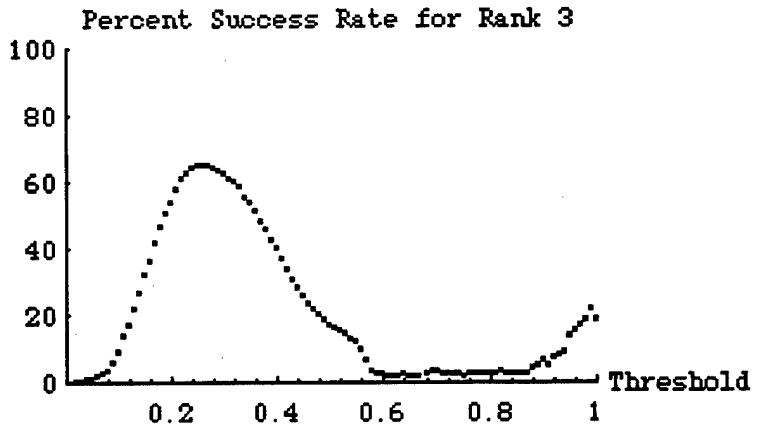

Figure 3. Percent success rates versus threshold for rank 3

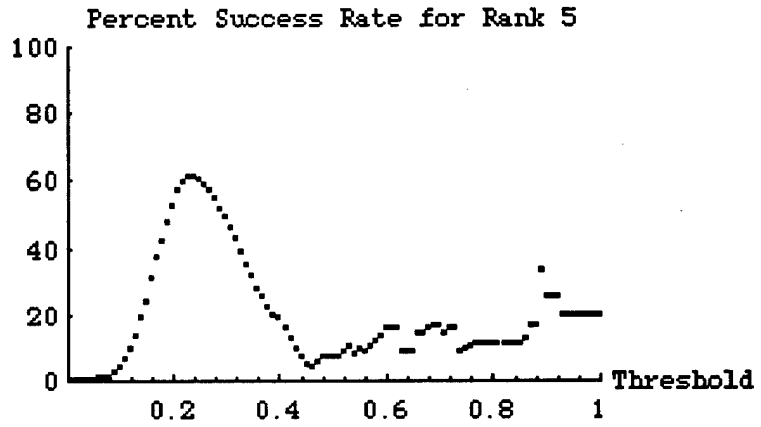

Figure 4. Percent success rates versus threshold for rank 5

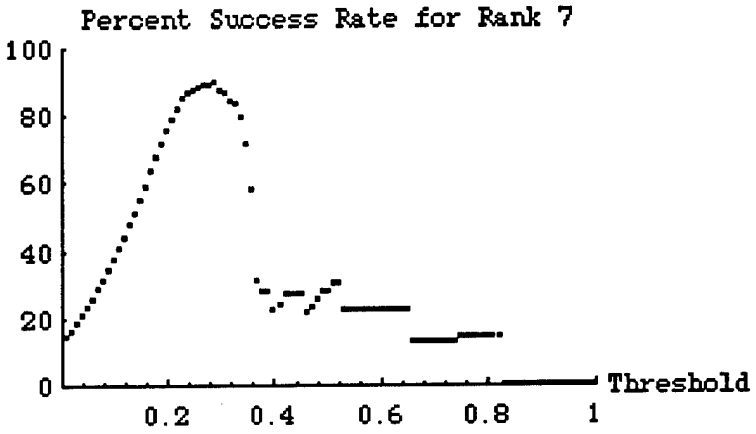

Figure 5. Percent success rates versus threshold for rank 7

The plots for ranks 2, 4 and 6 were similar to those for ranks 3, 5 and 7, with maxima between 0.2 and 0.3.

We then ran another case with the thresholds for P_2 through P_7 taken as the maxima from the above synchronized threshold plots. The maxima for P_2 through P_7 are the set {0.29, 0.25, 0.25, 0.23, 0.23, 0.29}. The threshold for P_1 was taken as the highest value (P_1 = 0.11) which still gave a 100% success rate for predicting rank 1. This is how we came up with the set of thresholds for P_1 through P_7 listed earlier, namely {0.11, 0.29, 0.25, 0.25, 0.23, 0.23, 0.29} .

C. Results Using the P_k Rank Profiles

The complete set of data for all seven profiles, using the thresholds determined in the previous section, is shown next.

Actual Rank	1	2	3	4	5	6	7	Total
Number of Matrices with Rank 1 profile	777	61	6	0	0	0	0	844
Number of Matrices with Rank 2 profile	142	676	63	14	1	0	0	896
Number of Matrices with Rank 3 profile	27	231	719	89	25	5	0	1096
Number of Matrices with Rank 4 profile	16	19	195	631	96	20	5	982
Number of Matrices with Rank 5 profile	10	6	14	252	683	111	58	1134
Number of Matrices with Rank 6 profile	17	6	1	12	192	839	559	1626
Number of Matrices with Rank 7 profile	11	1	2	2	3	25	378	422

Table 2. Raw data with the noise level set at 0.1

The data table shown above is for an (absolute) noise level of 0.1. The same data was generated for the following noise levels: 0.5, 1, 2, 3, 4, 5, 6, 8 and 10. For each noise level the percent success rates for each rank profile were computed. These percent success rates are listed in Table 3 which follows.

Noise Level	0.1	0.5	1.0	2.0	3.0	4.0	5.0	6.0	8.0	10.0
Percent Success Rate for Rank 1 profile	92	83	75	73	64*	25*	60*	60*	100*	33*
Percent Success Rate for Rank 2 profile	75	51	42	30	28	22*	20*	17*	18*	15*
Percent Success Rate for Rank 3 profile	66	44	34	26	21	18	16	17	16	15
Percent Success Rate for Rank 4 profile	64	37	28	22	19	19	15	21	14	15
Percent Success Rate for Rank 5 profile	60	37	27	22	22	18	16	16*	15	15
Percent Success Rate for Rank 6 profile	52	42	33	23	18	19	16	20	20	13
Percent Success Rate for Rank 7 profile	90	54	36	23	18	16	14	15	14	13

Table 3. Percent success rates for predicting the exact rank versus the noise level.

Some of these percentages were computed with less than 50 matrices in the sample. These numbers - listed with an asterisk next to them - are probably unreliable. Since there are seven possible ranks, choosing a rank randomly one would expect to be correct once in every seven guesses which is about 14 percent. Hence, 14 percent can be regarded as the background level.

The percent success rates were also computed using the rank profiles to predict the rank plus or minus one. For example, we computed the success rate using the profile for rank two when the

actual rank was one, two or three (that is, two plus or minus one). Using the numbers in Table 2. there were 142 matrices with an actual rank of one that satisfied the profile of rank two. There were 676 matrices with an actual rank of two that satisfied the profile of rank two, and there were 63 matrices with an actual rank of three that satisfied the profile of rank two. The total of these numbers is 881 matrices. The total of matrices from all seven actual ranks with the profile of rank two is 896. Therefore, dividing the 881 by 896 and multiplying by 100 percent, yields a success rate of 98 percent. Performing this calculation on all the other ranks for the various noise levels, we get Table 4.

Noise Level	0.1	0.5	1.0	2.0	3.0	4.0	5.0	6.0	8.0	10.0
Percent Success Rate for Rank 1 profile	99	95	94	88	96*	50*	60*	100*	100*	33*
Percent Success Rate for Rank 2 profile	98	94	91	89	73	64	62	56	45	45
Percent Success Rate for Rank 3 profile	95	84	78	68	57	54	48	47	47	44
Percent Success Rate for Rank 4 profile	94	79	72	59	52	51	47	52	48	43
Percent Success Rate for Rank 5 profile	92	79	68	57	57	51	48	42*	48	44
Percent Success Rate for Rank 6 profile	98	87	76	62	54	53	47	50	49	43
Percent Success Rate for Rank 7 profile	95	74	55	39	34	31	28	29	27	27

Table 4. Percent success rates for predicting the rank plus or minus one versus the noise level.

It should be noted that in Table 4 the end ranks of one and seven were treated somewhat differently. For rank one "plus or minus one" the minus one had no meaning; hence, only matrices with actual ranks of one and two with the profile of rank one were counted. Similarly, only matrices with actual ranks of six and seven with the profile of rank seven were counted. Again, the asterisk denotes cases where the success rate was computed using fewer than 50 matrices, and are statistically unreliable.

D. Comparison with a Singular Value Approach

For the pool of 7000 matrices used in the above analysis we also computed the singular values. This was done for comparison purposes. There are several methods for determining the effective rank of a matrix from its set of singular values. One such method involves using seven times the noise level as a threshold [1]. For example, if only three of the set of seven singular values for our 7x7 matrices are above this threshold, then the effective rank would be determined to be three. In fairness, the authors that suggested this threshold point out that there are other thresholds that would provide a somewhat reduced success rate over a larger range of noise levels. There are also several other singular value methods in vogue. The "3 db method", for example, starts with the largest singular value and tests successively smaller singular values to see if the square of one singular value is suddenly a factor of two or more less than the the square of the previous singular value. If this condition is discovered for the Nth singular value, then this Nth value and all smaller singular values are considered below the threshold.

Due to time considerations we only performed one comparison. We compared the success rates using our P_k profiles to the singular value threshold of seven times the noise level. We were especially interested in how our profile model compared with this singular value approach for low rank determination. Our statistics were poor for the larger noise values in the exact determination of rank one and rank two separately (the first two rows of Table 3.). Since our statistics were also poor for the rank plus or minus one case for data with the profile of rank one (the first row of Table 4.), we used the rank plus or minus one case for data with the profile of rank two (the second row of Table 4.). For comparison, we used those matrices with actual ranks of one, two and three that had only two singular values above the threshold and would therefore be predicted as rank two. We gathered this data for each of the noise levels. It is tabulated in Table 5 and plotted in Figure 6.

Noise Level	0.1	0.5	1.0	2.0	3.0	4.0	5.0	6.0	8.0	10.0
Percent Success Using Profiles	98	94	91	89	73	64	62	56	45	45
Percent Success Using Singular Values	100	97	90	62	40	26	19	14	10	10
Ratio of the Above Success Rates	0.98	0.97	1.01	1.44	1.83	2.46	3.26	4.00	4.09	4.50

Table 5. Percent success rates for rank two plus or minus one using our Pk model and the singular value model described above. Also, the ratio of these success rates is tabulated.

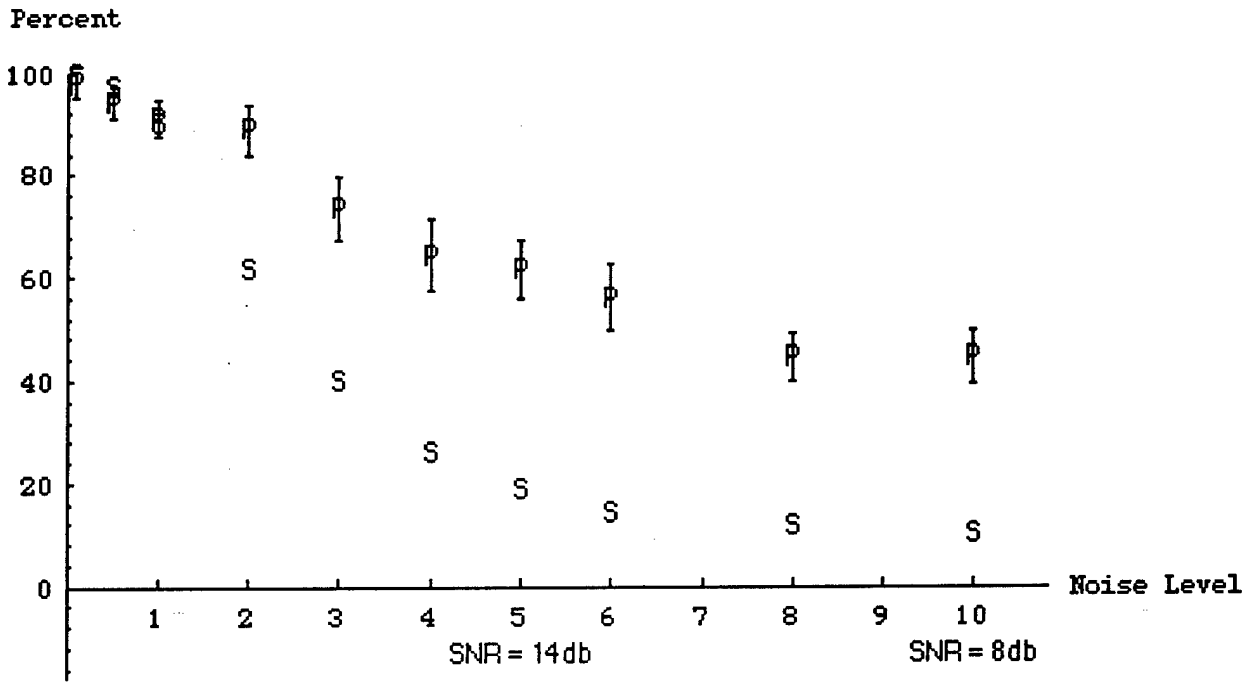

Figure 6. Percent success rates for rank two plus or minus one, using our profile model, the "P's", and the singular value model, the "S's", described below.

The error bars on the "P's" were determined using the following process. The sum of the numbers of matrices with actual ranks one, two and three which satisfied the profile of rank two was computed. Then the square root of this sum was added to and subtracted from this sum. Finally, these numbers were divided appropriately to become percentages. Using the same approach for the singular value analysis, the error bars on the "S" values were smaller than the "S" itself except when the noise level was less than one. When the noise was less than one, the error bars were only slightly larger than the "S". Because they were generally very small, the "S" error bars were omitted.

The signal to noise ratios were determined using the formula:

$$SNR = 10\log\left(\frac{\sum_{i=1}^{7}\lambda_i^2}{\sigma}\right) \qquad (4.1)$$

where SNR = the signal to noise ratio
σ = the noise level
λ_i = the ith eigenvalue

In our experiment the noise level σ was set at a particular value and 1000 matrices were generated for each of the seven ranks. The signal is t_2, that is, the sum of the squares of the eigenvalues (before the noise is added). Recall, the nonzero eigenvalues were allowed to independently range from negative thirty to positive thirty (excluding zero). Consequently, the signal, as measured by t_2, varied considerably over a set of 1000 matrices. During the experiment itself the eigenvalues were not recorded. The experimental data recorded in Table 5 (Figure 6) included matrices of actual rank one, two and three in different combinations depending on the noise level. Later, a large number of eigenvalues were generated using the same algorithm used during

the experiment. The value of t_2 was computed for sets of one, two and three of these eigenvalues. The numbers of t_2 values used for each of the three ranks was in the same proportion as the data in the experiment for noise at the level of 5 and 10. Average values and standard deviations were computed for the t_2 values. Using these average values and a one standard deviation variation, it was determined that when the noise level was 5, the SNR varied asymmetrically from 7 to 16 db with an average value of 14 db. When the noise level was 10, the SNR varied from 2 to 10 db with an average of 8 db.

Another way to view these results is to plot the ratio of the success rates, that is, the success rate using the P_k profile method divided by the success rate using the singular value method. This ratio versus the noise level is plotted in Figure 7. The error bars were determined using the uncertainties only for the profile success rate as discussed earlier. One can see that for noise levels less than or equal to one, the ratio is about one, and the success rates are about equal. For higher noise levels, the ratio gradually rises to about four and one-half. The profile method becomes about four and one-half times more successful than the tested singular value method for larger noise levels.

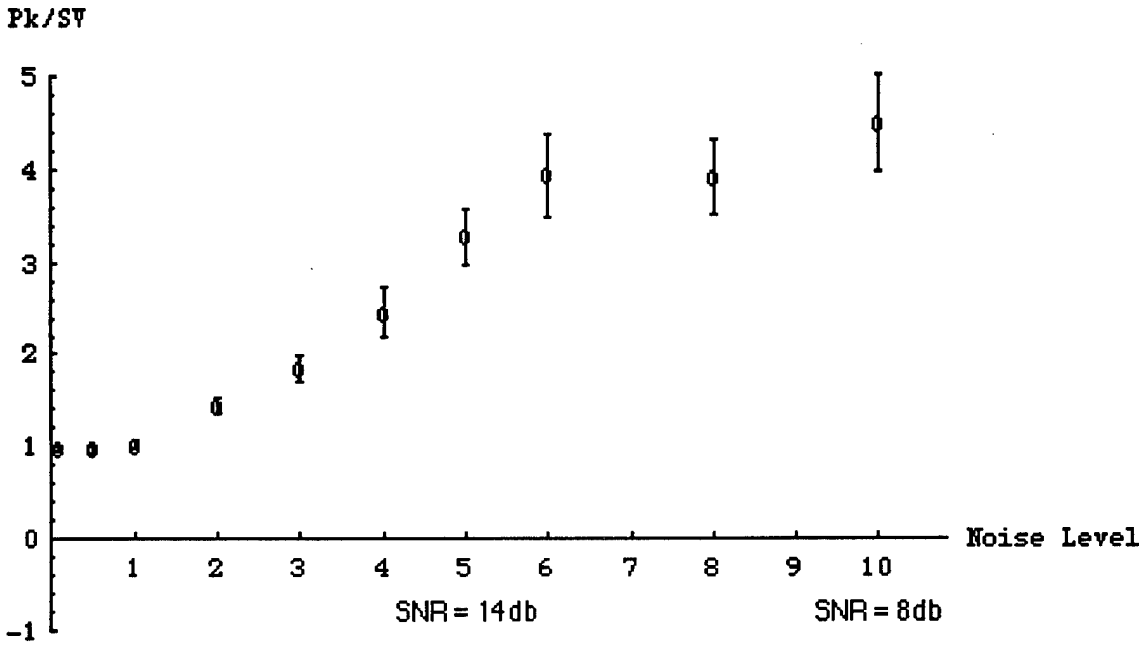

Figure 7. The ratio of the success rates (profile method divided by the singular value method) versus the noise level.

E. Additional Studies

Recall, that an earlier study had been carried out using percentage noise levels. With noise computed in this alternative manner we checked the effect of varying the range of the eigenvalues, and also examined the effects that occur when the eigenvalues become complex after the noise has been added.

1. Eigenvalue Range

When we made the range of the eigenvalues smaller, this improved our ability to distinguish ranks. We compared the P7 data at the one percent noise level to see how distinguishable rank 6 was from rank 7 for different ranges of eigenvalues. We generated 1000 matrices for each rank, and for comparison purposes used a threshold of P7 = 0.23 Using the usual range of eigenvalues, namely -30 to +30 (excluding 0), we found that 78% of the rank 6 matrices had their P7 below the threshold of 0.23, and 76% of the rank 7

matrices had their P7 above this threshold. When we shortened the range of eigenvalues to -10 to +10 (excluding 0), we found for rank 6, that there were 77% below the threshold; whereas, for rank 7, there were 87% above 0.23. The main difference is that when the range is smaller, a somewhat higher percentage of the rank 7 matrices is above the threshold.

Range of Eigenvalues	-30 to +30	-10 to +10
Rank 6 Below Threshold	78%	77%
Rank 7 Above Threshold	76%	87%

Table 6. Effect of eigenvalue range on the ability to distinguish P7 from ranks 6 and 7.

2. Complex eigenvalues

The process of adding noise to the matrices, changes the eigenvalues. In some cases the real eigenvalues become complex. In the earlier percentage noise studies we generated 3000 matrices for both ranks 6 and 7 with the noise level set at one percent. For the rank 6 matrices 2117 of the 3000 matrices (71%) still had all their eigenvalues real; however, the remaining 883 matrices (29%) had at least one pair of complex eigenvalues. For the rank 7 matrices 2092 (70%) had all the eigenvalues remain real, while the remaining 909 (30%) had at least one pair of complex eigenvalues.

When we examined how well the ranks were separated by a P7 threshold of 0.23, we found that when the eigenvalues stayed real we were better able to distinguish the P7 values of rank 6 from those of rank 7. Specifically, when the eigenvalues stayed real, 91% of the rank 6 matrices were below the 0.23 threshold, and 73% of the rank 7 matrices were above this threshold. However, for the matrices that had at least one pair of complex eigenvalues, only 46% of the rank 6 matrices were below the threshold, and 74% of the rank 7 matrices were above the threshold. A much smaller

percentage of the values of P7 for rank 6 are less than the threshold when the eigenvalues are complex than when the eigenvalues stay real.

Nature of Eigenvalues	All real	At least one complex pair
Rank 6 Below Threshold	91%	46%
Rank 7 Above Threshold	73%	74%

Table 7. Effect of real versus complex eigenvalues on the ability to distinguish P7 from ranks 6 and 7.

Finally, we found a correlation between matrices which had complex eigenvalues after the noise was added and the determinant of the similarity transformation. Of course, the eigenvalues after the similarity transformation (and before the noise was added) were the same as the eigenvalues of the original (diagonal) matrix. However, when noise was added at the one percent level, matrices that had been created by similarity transformation matrices with very small determinants were significantly more likely to have complex eigenvalues than those created with similarity transformation matrices with larger determinants. Of 19 matrices studied whose similarity transformations had determinants with absolute values less than 0.01, we found 18 of them had complex eigenvalues after the noise was added. The remaining matrix's eigenvalues stayed real, but they were changed substantially. For a control group, we studied 18 matrices whose similarity transformations had determinants with absolute values greater than 1.0. In this set only two had complex eigenvalues.

V. Summary and Conclusion

Assuming there are ways to estimate the uncertainty in the matrix elements, the P_k profile of a given matrix can be used to determine its rank.

Most of the analysis is performed ahead of time. This prior analysis includes a Monte Carlo generation of matrices with the assumed known uncertainty level and spectrum of eigenvalues. This Monte Carlo simulation is used to generate threshold levels for each of the P_k's. It is also used to compute a table of rank probabilities for each of the possible P_k profiles. At run time the P_k's of the given matrix are computed and compared to these threshold levels. This comparison provides the P_k profile. This matrix P_k profile, along with the probability table resulting from the Monte Carlo simulation, can then be used to compute the probability of every rank. The rank with the highest probability is then predicted as the rank of the given matrix.

The P_k's are computed from the coefficients of the characteristic polynomial (the C_k's). These C_k's in turn can be determined from polynomials involving the elements of the original matrix. This has the potential for being faster than iterative approaches, for example methods using singular values.

Moreover, we have shown that this P_k profile approach compares very favorably with at least one singular value analysis. As the noise level was increased, we found that the P_k profile method gradually became more successful at predicting the rank than the tested singular value approach.

Future work in this area includes trying to find faster algorithms for computing the C_k's. Also, the efficiency of using only the P_k thresholds with small values of k to determine low ranks should be examined. Currently, if the rank is one, the P_k values for the largest values of k are computed first to eliminate the possibility of the highest ranks. However, the P_k values with the largest k values take the longest to compute. While in principle it is important to perform the analysis in this top down manner, it is not

clear how much would be lost if the run time analysis were performed in a bottom up manner, if the intention is to detect low rank matrices.

Acknowledgements

We would like to express our appreciation to Professor Michael Chamberlain (U.S. Naval Academy) who participated in the early stages of the development of this model and who wrote much of the computer software used in the analysis. Also, we would like to thank Professor Peter Turner (U.S. Naval Academy) for his valuable comments on an early draft of this paper.

Reference

[1] Konstantinides, K., "Statistical Analysis of Effective Singular Values in Matrix Rank Determination", IEEE Transaction on Acoustics, Speech, and Signal Processing, Vol. 36, pp. 757-763, (May, 1988).

Distribution List

Report No. NAWCADWAR- -96 - 21 - TR

	No. of Copies
Office of Naval Research 800 N. Quincy St. Arlington, VA 22217	2
Marine Corps Research Center 2040 Broadway Street Quantico, VA 22134-5107	
Marine Corps University Libraries	2
Naval Air Systems Command Air-5002 Washington, DC 20641-5004	
Technical Information & Reference Center	2
Naval Air Warfare Center, Aircraft Division Building 407 Patuxent River, MD 20670-5407	
Naval Air Station Central Library	2
Naval Air Systems Command (NAVAIR) Jefferson Plaza Bldg 1., 1421 Jefferson Davis Hwy Arlington, VA 2243-5120	
Director Science & Technology (4.0T)	2
Naval Sea Systems Command 2531 Jefferson Davis Hwy Arlington, VA 22242-5100	
Technical Library, (SEA04TD2L)	2
Defense Technical Information Center Cameron Station BG5 Alexandria, VA 22304-6145	
DTIC-FDAB	2
U.S. Naval Academy Annapolis, MD 21402-5029	
Dr. Richard Werking (Nimitz Library)	2
Naval Air Warfare Center Weapons Division China Lake, CA 93555-6001	
Head Research & Tech. Div. (NAWCWPNS-474000D)	2
Computational Sciences (NAWCWPNS-474400D)	2
Mary-Deirdre Coraggio (Library Division, C643)	2
Trenton State College Trenton, NJ 08650	
Ronald F. Gleeson (Department of Physics)	5

Distribution List, cont.

Report No. NAWCADWAR- -96- 21 - TR

	No. of Copies
Naval Postgraduate School Monterey, CA 93943-5002	
Dudley Knox Library	2
Texas A & M University College Station, TX 77843-3368	
Peter F. Stiller (Department of Mathematics)	5
Naval Research Laboratory(NRL) 4555 Overlook Ave, SW Washington, DC 20375-5000	
Center for Computational Science (NRL-5590)	2
Superint., Lab. for Comput. Phy & Fluid Dynamics (NRL-6400)	2
Ruth H. Hooker Research Library (5220)	2
Naval Command, Control & Ocean Surveillance Center 200 Catalina Blvd San Diego, CA 92147-5042	
Technical Library (NRAD-0274)	2
Signals Warfare Div (NRAD-77)	2
Analysis & Simulation Div. (NRAD-78)	2
Director of Navigation & Air C3 Dept. (NCCOSC-30)	2
Naval Air Warfare Center Aircraft Division Warminster Warminster, PA 18974-0591	
Warfare Planning Systems (4.5.2.1.00R07)	2
Tactical Inf. Systems (4.5.2.2.00R07)	2
Mission Comp. Processors (4.5.5.1.00R07)	2
Dr. Robert M. Williams (4.5.5.1.00R07)	20
Acoustic Sensors (4.5.5.4.00R07)	2
RF Sensors (4.5.5.5.00R07)	2
EO Sensors (4.5.5.6.00R07)	2
Inductive Analysis Branch (4.10.2.00R86)	2
TACAIR Analysis Division (4.10.1.00R86)	2
Operations Research Analysis Branch (4.10.1.00R86)	2
Advanced Concepts Branch (4.10.3.00R86)	2
Nav. Aval. Sys. Dev. Division (3.1.0.9)	2
Anthony Passamante (4.5.5.3.4.00R07)	2
Elect. Systems BR (4.8.2.2.00R08)	2
Dr. Richard Llorens (4.3.2.1.00R08)	2
Advanced Processors (4.5.5.1.00R07)	2
Mission & Sensors Integrations (4.5.5.3.000R07)	2
Applied Signal Process BR (4.5.5.3.4.00R07)	2